Pattern Recognition, Vol. 25, No. 9, pp. 975–985, 1992
Printed in Great Britain

0031-3203/92 $5.00 + .00
Pergamon Press Ltd
Pattern Recognition Society

ROBUST POSITION, SCALE, AND ROTATION INVARIANT OBJECT RECOGNITION USING HIGHER-ORDER NEURAL NETWORKS

LILLY SPIRKOVSKA and MAX B. REID
NASA Ames Research Center, M/S 269-3, Moffett Field, CA 94035-1000, U.S.A.

(*Received* 17 *June* 1991; *in revised form* 17 *December* 1991; *received for publication* 24 *January* 1992)

Abstract—For object recognition invariant to changes in the object's position, size, and in-plane rotation, higher-order neural networks (HONNs) have numerous advantages over other neural network approaches. Because distortion invariance can be built into the architecture of the network, HONNs need to be trained on just one view of each object, not numerous distorted views, reducing the training time significantly. Further, 100% accuracy can be guaranteed for noise-free test images characterized by the built-in distortions. Specifically, a third-order neural network trained on just one view of an SR-71 aircraft and a U-2 aircraft in a 127 × 127 pixel input field successfully recognized all views of both aircraft larger than 70% of the original size, regardless of orientation or position of the test image. Training required just six passes. In contrast, other neural network approaches require thousands of passes through a training set consisting of a much larger number of training images and typically achieve only 80–90% accuracy on novel views of the objects. The above results assume a *noise-free* environment. The performance of HONNs is explored with non-ideal test images characterized by white Gaussian noise or partial occlusion. With white noise added to images with an ideal separation of background vs. foreground gray levels, it is shown that HONNs achieve 100% recognition accuracy for the test set for a standard deviation up to ~10% of the maximum gray value and continue to show good performance (defined as better than 75% accuracy) up to a standard deviation of ~14%. HONNs are also robust with respect to partial occlusion. For the test set of training images with very similar profiles, HONNs achieve 100% recognition accuracy for one occlusion of ~13% of the input field size and four occlusions of ~70% of the input field size. They show good performance for one occlusion of ~23% of the input field size or four occlusions of ~15% of the input field size each. For training images with very different profiles, HONNs achieve 100% recognition accuracy for the test set for up to four occlusions of ~2% of the input field size and continue to show good performance for up to four occlusions of ~23% of the input field size each.

Neural networks Higher-order White noise Gaussian noise Occlusion
Object recognition Invariant classification Coarse-coding

INTRODUCTION

An important aspect of the human visual system is the ability to recognize an object despite changes in the object's position in the input field, its size, or its orientation. For many industrial applications, machine vision systems must also have this ability. Traditionally, machine vision systems have separated the object recognition task into two independent subtasks: feature extraction followed by classification.[1,2] The feature extraction task begins with an object and a procedure for extracting relevant features. The features are chosen by the system designer to be invariant to the object's position, scale, and orientation. The output of this task, which is a vector of feature values, is then passed to the classification subtask. Based on a large set of these vectors, the classifier determines which are the distinguishing features of each object class such that new vectors are placed into the correct class within a predetermined error.

A more recent approach for distortion invariant object recognition combines the tasks into a single system. Given only a set of views of each object class it is required to distinguish between, the system determines which features to extract as well as which are the distinguishing features of each class. The advantage of this approach is that the two subtasks can share information and improve the classifier's separating ability by extracting the useful features. The disadvantage, however, is that the system requires a longer training period since it has no prior information about the relationship between the set of training views. Object recognition systems based on neural networks are an example of this approach.

The most popular method used in neural network-based object recognition systems is backpropagation-trained first-order neural networks,[3] as illustrated in Fig. 1. The training process consists of applying input vectors (distorted views of each class) sequentially and adjusting the network weights using a gradient descent learning rule until the input vectors produce the desired output vectors (class assignment) within some predetermined error. For a first-order network to learn to distinguish between a set of objects independent of their position, scale, or in-plane orientation, the network must be trained on a large set of distorted views. The desired effect of including distorted views into the training set is that the hidden layers will extract the necessary invariant features and the network will generalize the input vectors so that it can also recognize

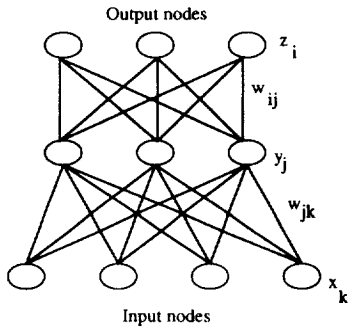

Fig. 1. First-order neural network. In a first-order neural network, input nodes are connected directly to output or hidden layer nodes. No advantage is taken of any known relationships between the input nodes.

distorted views that are not part of the training set. Such generalization has been demonstrated in numerous simulations. However, because first-order networks do not take advantage of predefined relationships between the input nodes, they require a large number of training passes to generalize the concepts behind the distortions. Also, even after extensive training with a large training set, they usually achieve only 80–90% recognition accuracy on novel examples.[3,4]

In order to eliminate these disadvantages, higher-order neural networks (HONNs) can be used.[5–7] HONNs incorporate domain-specific knowledge into a single network. In the position, scale, and rotation-invariant object recognition domain, distortion invariance can be built directly into the architecture of the network and does not need to be learned. Thus, the network needs to be trained on just one view of each object, not numerous distorted views, reducing the training time significantly. Moreover, 100% recognition accuracy is guaranteed for noise-free images characterized by the built-in distortions.

In the next section we will discuss how known relationships can be exploited and desired invariances built into the architecture of HONNs, explain the limitations of using HONNs with higher resolution images, and describe how coarse coding can be applied to HONNs to increase the input field size for use with practical object recognition problems. We will then present details on the performance of third-order networks on non-ideal images characterized by white Gaussian noise or partial occlusion.

HIGHER-ORDER NEURAL NETWORKS

The output of a node, denoted by y_i for node i, in a general higher-order neural network is given by

$$y_i = \Theta(\Sigma_j w_{ij} x_j + \Sigma_j \Sigma_k w_{ijk} x_j x_k + \Sigma_j \Sigma_k \Sigma_l w_{ijkl} x_j x_k x_l + \cdots) \quad (1)$$

where $\Theta(f)$ is a non-linear threshold function such as the hard limiting transfer function given by

$$y_i = 1, \quad \text{if} \quad f > 0 \quad (2)$$
$$y_i = 0, \quad \text{otherwise}$$

the values of x are the excitation values of the input nodes; and the interconnection matrix elements, w, determine the weight that each input is given in the summation. The interconnection weights can be constrained such that invariance to given distortions is built directly into the network architecture.

For instance, consider a second-order network as illustrated in Fig. 2. In a second-order network, the inputs are first combined in pairs and then the output is determined from a weighted sum of these products. The output for a strictly second-order network is given by the function

$$y_i = \Theta(\Sigma_j \Sigma_k w_{ijk} x_j x_k). \quad (3)$$

The invariances achieved using this architecture depend on the constraints placed on the weights.

As an example, each pair of input pixels combined in a second-order network define a line with a certain slope. As shown in Fig. 3, when an object is moved or scaled, the two points in the same relative position within the object still form the endpoints of a line with the same slope. Thus, provided that all pairs of points which define the same slope are connected to the output node using the same weight, the network will be invariant to distortions in scale and translation. In particular, for two pairs of pixels (j, k) and (l, m), with coordinates (x_j, y_j), (x_k, y_k), (x_l, y_l), and (x_m, y_m), respectively, the

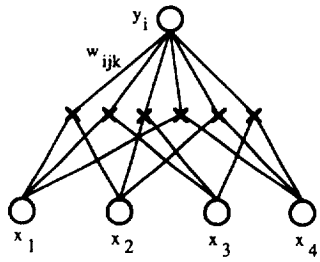

Fig. 2. Second-order neural network. In a second-order neural network, the inputs are first combined in pairs (at X) and the output is determined from a weighted sum of these products.

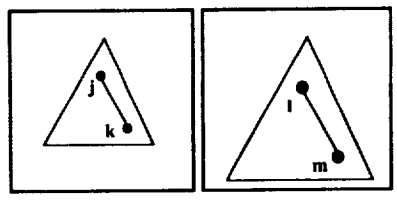

Fig. 3. Translation and scale invariance in a second-order network. By constraining the network such that all pairs of points which define equal slopes use equal weights, translation and scale invariance are incorporated into a second-order neural network.

weights are constrained according to

$$w_{ijk} = w_{ilm}, \text{ if } (y_k - y_j)/(x_k - x_j) = (y_m - y_l)/(x_m - x_l). \tag{4}$$

Alternatively, the pair of points combined in a second-order network may define a distance. As shown in Fig. 4, when an object is moved or rotated within a plane, the distance between a pair of points in the same relative position on the object does not change. Thus, as long as all pairs of points which are separated by equal distances are connected to the output with the same weight, the network will be invariant to translation and in-plane rotation distortions. The weights for this set of invariances are constrained according to

$$w_{ijk} = w_{ilm}, \text{ if } \|\mathbf{d}_{jk}\| = \|\mathbf{d}_{lm}\|. \tag{5}$$

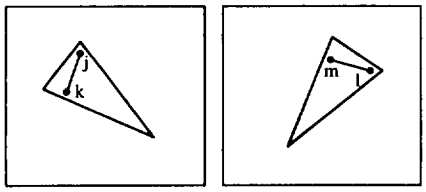

Fig. 4. Translation and rotation invariance in a second-order network. By constraining the network such that all pairs of points which are equal distances away use equal weights, translation and rotation invariances are incorporated into a second-order network.

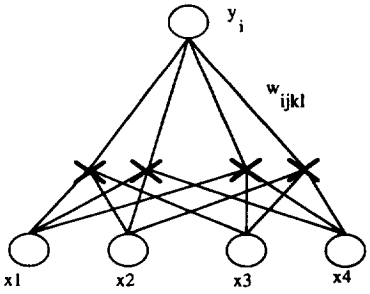

Fig. 5. Third-order neural network. In a third-order neural network, input nodes are first multiplied together in triplets (at X) and then the output is determined from a weighted sum of the products.

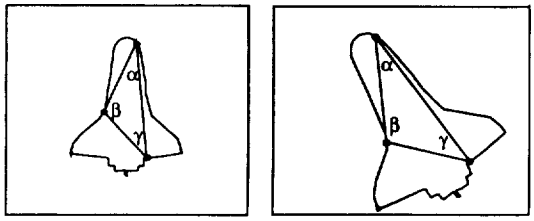

Fig. 6. Position, scale, and rotation invariance in a third-order network. As long as all similar triangles are connected to the output with the same weight, a third-order network will be invariant to scale, in-plane rotation, and translation distortions.

That is, the magnitude of the vector defined by pixels j and $k(\mathbf{d}_{jk})$ is equal to the magnitude of the vector defined by pixels l and $m(\mathbf{d}_{lm})$.

To achieve invariance to translation, scale, and in-plane rotation simultaneously, a third-order network can be used. The output for a strictly third-order network, shown in Fig. 5, is given by the function

$$y_i = \Theta(\Sigma_j \Sigma_k \Sigma_l w_{ijkl} x_j x_k x_l). \tag{6}$$

All sets of input pixel triplets are used to form triangles with included angles (α, β, γ) as shown in Fig. 6. When the object is translated, scaled, or rotated in-plane, the three points in the same relative positions on the object still form the included angles (α, β, γ). In order to achieve invariance to all three distortions, all sets of triplets forming similar triangles are connected to the output with the same weight. That is, the weight for the triplet of inputs (j, k, l) is constrained to be a function of the associated included angles (α, β, γ) such that all elements of the alternating group on three elements (group A3) are equal:

$$w_{ijkl} = w(i, \alpha, \beta, \gamma) = w(i, \beta, \gamma, \alpha) = w(i, \gamma, \alpha, \beta). \tag{7}$$

Note that the order of the angles matters but not which angle is measured first.

The constraint that all similar triangles are connected to the output node with the same weight can be implemented in two steps:

(a) Calculate the included angles α, β, and γ (to some granularity) formed by each combination of three pixels for a given input field size. Since this computation is expensive and the combination of triplets for a given input field size does not depend on the objects to be distinguished, these angles can be precalculated and stored. This step would then be modified to read the included angles corresponding to each combination of three pixels from a file rather than calculating them.

(b) Set up the correspondence between the angles α, β, and γ (same granularity) such that all triplets of the angles which are members of the alternating group (that is, the order of the angles matters, but not which one comes first) point to a single memory location. This assures that all similar triangles will manipulate the same weight value. Our implementation uses three matrices (*w*, *w_angle*, and *w_invar*) linked with pointers. Each location in *w* (indexed by the triple i, j, k representing the input pixels) points to a location in *w_angle* (indexed by the triple α, β, γ representing the angles formed by the triple *ijk*). Similarly, each location in *w_angle* points to a location in *w_invar*, also indexed by a triple of angles α, β, γ ordered such that the smallest angle is assigned to α. That is, *w_angle*[80][60][40] points to *w_invar*[40][80][60], as do the elements *w_angle*[60][40][80] and *w_angle*[40][80][60].

The implementation of the second-order network constraints specified by equation (4) or equation (5) requires obvious modification to the above steps.

Because HONNs are capable of providing non-linear separation using only a single layer, once invariances

are incorporated into the architecture, the weights can be modified using a simple rule of the form

$$\Delta w_{ijk} = (t_i - y_i)x_j x_k \tag{8}$$

for a second-order network, or

$$\Delta w_{ijkl} = (t_i - y_i)x_j x_k x_l, \tag{9}$$

for a third-order network, where the expected training output, t, the actual output, y, and the inputs, x, are all binary.

The main advantage of building invariance to geometric distortions directly into the architecture of the network is that the network is forced to treat all distorted views of an object as the same object. Distortion invariance is achieved before any input vectors are presented to the network. Thus, the network needs to learn to distinguish between just one view of each object, not numerous distorted views.

The most severe limitation of this method is that the number of possible triplet combinations increases as the size of the input field increases. In an $N \times N$ pixel input field, combinations of three pixels can be chosen in N^2-choose-3 ways. Thus, for a 9×9 pixel input field, the number of possible triplet combinations is 81-choose-3 or 85,320. Increasing the resolution to 128×128 pixels increases the number of possible interconnections to 128^2-choose-3 or 7.3×10^{11}, a number too great to store on most machines. On our Sun 3/60 with 30 MB of swap space, we can store a maximum of 5.6 million (integer) interconnections, limiting the input field size for fully connected third-order networks to 18×18 pixels.

To circumvent this limitation, we use a coarse coding algorithm which allows a third-order network to be used with a practical input field size of at least 127×127 pixels while retaining its ability to recognize images which have been scaled, translated, or rotated in-plane. Training takes just a few passes and training time is on the order of minutes on a Sun 3/60 or less than a second on a Sun SPARCstation 2.

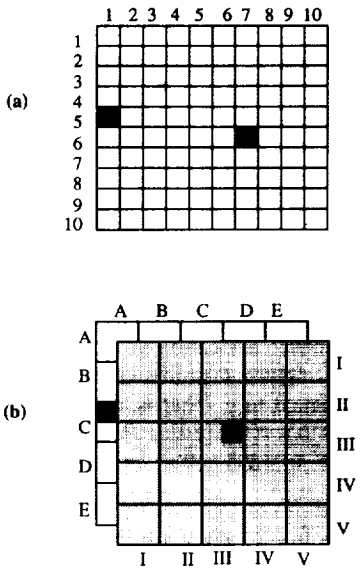

Fig. 7. An example of a coarse-coded input field: (a) a 10×10 pixel input field; (b) two fields of 5×5 coarse pixels.

The coarse coding representation we use involves overlaying fields of coarser pixels in order to represent an input field composed of smaller pixels,[8,9] as shown in Fig. 7. Figure 7(a) shows an input field of size 10×10 pixels. In Fig. 7(b), we show two offset but overlapping fields, each of size 5×5 "coarse" pixels. In this case, each coarse field is composed of pixels which are twice as large (in both dimensions) as in Fig. 7(a). To reference an input pixel using the two coarse fields requires two sets of coordinates. For instance, the pixel $(x = 7, y = 6)$ on the original image would be referenced as the set of coarse pixels $((x = D, y = C)$ and $(x = \text{III}, y = \text{III}))$, assuming a coordinate system of (A, B, C, D, E) for coarse field one and (I, II, III, IV, V) for coarse field two. In order to represent pixels which are not intersected by all coarse fields, such as pixel (1, 5), coarse coding can be implemented either by using wraparound or by using only the intersection of the fields. If coarse coding is implemented using wraparound, pixel (1, 5) could be represented as the set of coarse pixels $((A, C)$ and $(\text{V}, \text{II}))$. On the other hand, if coarse coding is implemented as the intersection of the coarser fields, the two fields shown in Fig. 7(b) would be able to uniquely describe an input field of 9×9 pixels, not 10×10.

Using wraparound, the relationship between the number of coarse fields (n), input field size (IFS), and coarse field size (CFS) in each dimension is given by

$$\text{IFS} = (\text{CFS} * n). \tag{10}$$

On the other hand, using the intersection of fields implementation, the relationship between number of coarse fields, input field size, and coarse field size in each dimension is given by

$$\text{IFS} = (\text{CFS} * n) - (n - 1). \tag{11}$$

The effective input field size, IFS, is not significantly different with either implementation for small n. Further, either implementation yields a one-to-one transformation. That is, each pixel on the original image can be represented by a unique set of coarse pixels.

The above transformation of an image to a set of smaller images can be used to greatly increase the resolution possible in a higher-order neural network. For example, a fully connected third-order network for a 100×100 pixel input field requires 100^2-choose-3 or 1.6×10^{11} interconnections. Using 10 fields of 10×10 coarse pixels requires just 10^2-choose-3 or 161,700 interconnections, accessed once for each field. The number of required interconnections is reduced by a factor of $\sim 100{,}000$.

As an example of how coarse coding can be applied to HONNs, consider training the network to distinguish between a "T" and a "C" in an 8×8 pixel input field, as shown in Fig. 8. With coarse coding implemented with wraparound, as explained previously, there are two possible combinations which will provide an

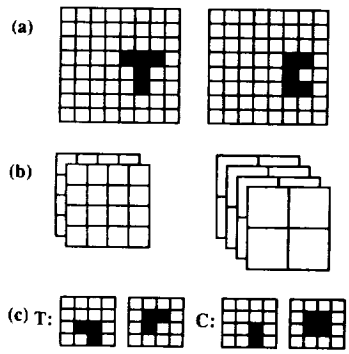

Fig. 8. Using coarse-coded fields with HONNs: (a) two training images in an 8 × 8 pixel input field; (b) two possible configurations of coarse pixels to represent the input field in (a); (c) coarse-coded representation of the training images in (a) using two layers of 4 × 4 coarse pixels.

effective input field of 8 × 8 pixels: two fields of 4 × 4 coarse pixels or four fields of 2 × 2 pixels. Both possibilities are shown in Fig. 8(b).

Applying coarse coding by using two fields of 4 × 4 coarse pixels, the two images shown in Fig. 8(a) are transformed into the four images shown in Fig. 8(c). Training of the network then proceeds in the usual way with one modification: the transfer function thresholds the value obtained from summing the weighted triangles over *all* coarse images associated with each training object. That is

$$y = 1, \quad \text{if } \{\Sigma_n(\Sigma_j\Sigma_k\Sigma_l w_{jkl}x_jx_kx_l)\} > 0$$
$$y = 0, \text{ otherwise} \qquad (12)$$

where j, k, and l range from one to the coarse pixel size squared, n ranges from one to the number of coarse fields, the values of x represent coarse pixel values, and w_{jkl} represents the weight associated with the triplet of inputs (j, k, l).

A more detailed analysis of coarse coding and its applicability with respect to higher-order neural networks is presented in reference (10). Here we present just a brief synopsis of its limitations including the minimum and maximum coarse field size, the minimum and maximum number of fields which can be used and still achieve position, scale, and rotation invariant recognition, and the maximum input field resolution possible.

We evaluated the coarse coding technique using the expanded version of the T/C problem. As explained in Rumelhart *et al.*[3] in the T/C problem both objects are constructed of five squares, as illustrated in Fig. 9, and the problem is to discriminate between them independent of translation or 90° rotations. In our work, the network was also required to distinguish between the objects invariant to distortions in scale.

The minimum possible coarse field size is determined by the training images. The network is unable to distinguish between the training images when the size of each coarse pixel is increased to the point where the training images no longer produce unique coarse coded

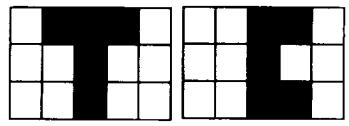

Fig. 9. T/C problem. In the T/C recognition problem, each pattern consists of five squares. Over all in-plane rotations and translations, the patterns can be discriminated only if combinations of triplets of pixels are examined.

representations. As an example, for the T/C recognition problem the minimum coarse field size which still produces unique representations is 3 × 3 pixels.

In contrast, the maximum limit is determined by the HONN architecture and the memory available for its implementation, and not by the coarse coding technique itself. As discussed previously, the number of possible triplet combinations in a third-order network is N^2-choose-3 for an $N \times N$ pixel input field and given the memory constraints of our Sun 3/60, the maximum possible coarse field size is 18 × 18 pixels.

Regarding the number of coarse fields which can be used and still achieve PSRI object recognition, the minimum is one field whereas the maximum has not yet been reached. A minimum of one coarse field represents the non-coarse-coded HONN case. In order to determine the limit for the maximum number of fields possible, we ran simulations on the T/C problem coded with a variable number of 3 × 3 coarse pixels. A third-order network was able to learn to distinguish between the two characters in less than ten passes in an input field size of up to 4095 × 4095 pixels using 2047 fields.† Increasing the number of fields beyond this was not attempted because 4096 × 4096 is the maximum resolution available on most image processing hardware which would be used in a complete HONN-based vision system. Also, each object in such a large field requires 16 MB of storage space. It takes only a few such objects to fill up a disk.‡

Finally, as with the maximum number of coarse fields, the maximum input field resolution possible with coarse coded HONNs has not been delimited. As discussed above, we trained a third-order network on the T/C problem in up to a 4096 × 4096 pixel input field. We expect a resolution of 4096 × 4096 is sufficient for most object recognition tasks. Notwithstanding, we also expect a greater resolution is possible.

2D OBJECT RECOGNITION SIMULATION RESULTS

We evaluated the performance of HONNs on numerous object recognition problems, including an SR-71/U-2 discrimination problem and an SR-71/Space

† An input field resolution of 4096 × 4096 was also achieved by using 273 fields of 16 × 16 coarse pixels.

‡ Note that this is not a limitation of the coarse-coding scheme itself. If a 4096 × 4096 pixel image could be stored on disk, using the intersection of fields approach to coarse coding, we could represent it as 228 fields of 18 × 18 coarse pixels, requiring only 5.6 MB memory.

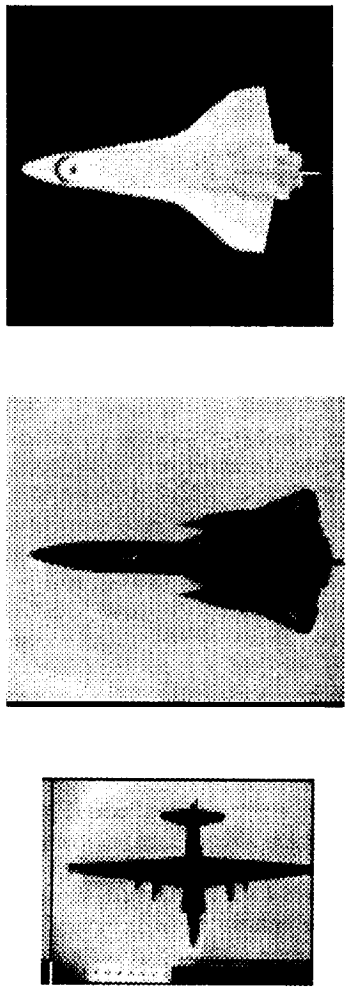

Fig. 10. The 8-bit gray level images from which the training images are generated: (a) Space Shuttle; (b) SR-71; (c) U-2.

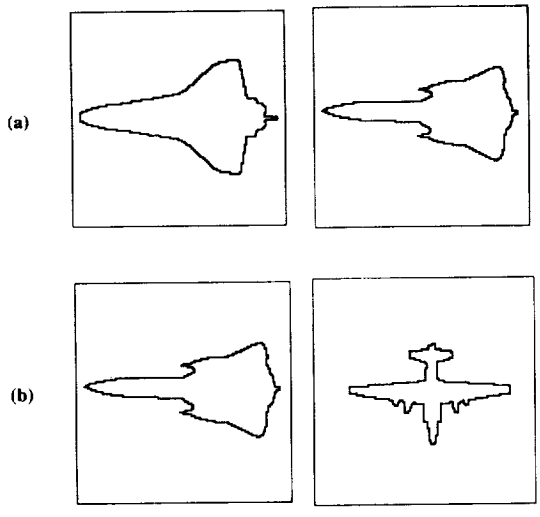

Fig. 11. Training images. One binary edge-only view each for: (a) Space Shuttle Orbiter vs. SR-71; (b) SR-71 vs. U-2.

Shuttle discrimination problem. The simulations used a coarse-coded third-order network designed for a 127×127 pixel input field. For ease of implementation, we used the intersection of fields approach to coarse coding with nine fields of 15×15 coarse pixels. The training sets were generated from actual models of the aircraft. The 8-bit gray level images of the aircraft are shown in Fig. 10. The images were thresholded to produce binary images, and then edge detected using a digital Laplacian convolution filter with a positive derivative to produce the images shown in Fig. 11. For rotated and scaled views of the objects, the original gray level images were first scaled, then rotated, and then thresholded and edge detected. Notice that the profiles of the SR-71 and Space Shuttle are somewhat similar whereas those of the SR-71 and U-2 are very different.

For both recognition problems, the network learned to distinguish between the aircraft in less than 10 passes through the training set.† The networks were then tested on a set of rotated, scaled, and translated views. A complete test set of translated, scaled, and 1° rotated views of the two objects in a 127×127 pixel input field consists of over 100 million images. Assuming a test rate of 200 images per hour, it would take computer-years to test all possible views. Accordingly, we limited the testing to a representative subset consisting of four sets:

(1) All translated views, but with the same orientation and scale as the training images.

(2) All views rotated in-plane at 1° intervals, centered approximately at the same position and of the same size as the training images.

(3) All scaled views of the objects, in the same orientation and centered at the same position as the training images.

(4) A representative subset of approximately 100 simultaneously translated, rotated, and scaled views of the two objects.

For both recognition problems, the networks achieved 100% accuracy on all test images in sets (1) and (2). Furthermore, the networks recognized, with 100% accuracy, all scaled views, from test set (3), down to 70% of the original size. Finally, for test set (4), the network correctly recognized all images larger than 70% of the original size, regardless of the orientation or position of the test image.

As shown in previous research,[7,10] invariance to scale is affected by the resolution to which the angles α, β, and γ in equation (7) are calculated. Briefly, as the resolution of the input field is increased, the resolution to which α, β, and γ are calculated can also be increased, generally increasing scale invariance. Scale invariance also varies with the coarse field size and the number of

† The SR-71 vs. Space Shuttle problem required two passes through the training set, while the SR-71 vs. U-2 required six passes.

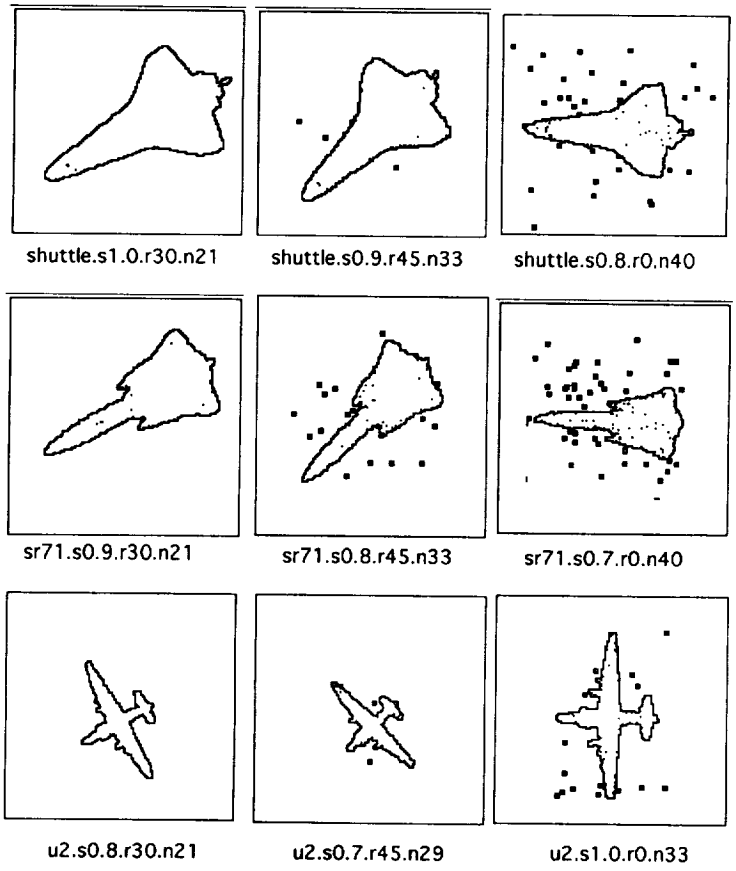

Fig. 12. Typical images used to test the tolerance of HONNs to white Gaussian noise. Test images were generated automatically by adding a normally distributed random gray value (with a mean of 0 and a standard deviation from 1 to 50) to the original gray level value and then binarizing and edge detecting the resulting image. Values for scale, rotation, and standard deviation are shown under each image. For example, "s0.7.r30.n20" is an image of scale 70%, rotation 30° and standard deviation of 20. For each object, we show a noise level recognized with an accuracy of 100, 75, and 50%.

coarse fields used. In general, a larger coarse field size yields greater scale invariance. However, the learning time also increases as the coarse field size increases. Thus, if less scale invariance can be tolerated, a desired input field size can be represented with a smaller coarse field size and greater number of coarse fields. A more detailed analysis of coarse coding and its applicability and limitations with respect to HONNs is presented in reference (10).

NOISE TOLERANCE SIMULATIONS

We evaluated the performance of HONNs with noisy images on the same two object recognition problems discussed above: the SR-71/U-2 discrimination problem and the SR-71/Space Shuttle discrimination problem. In particular, following training, the networks were tested on numerous non-ideal images in order to determine the performance of HONNs to white Gaussian noise and partial occlusion.

WHITE GAUSSIAN NOISE

To test the tolerance of HONNs to white noise, each instantiation of the network (one for the SR-71/U-2 problem and one for the Shuttle/SR-71 problem) was tested on 1200 images generated by modifying the 8-bit gray level values of the original images using a Gaussian distribution of random numbers with a mean of 0 and a standard deviation of between 1 and 50. The test set consisted of 50 images (with increasing standard deviation from 1 to 50)† for each of four scales ranging from 70 to 100% in 10% increments, and each rotation of 0°, 30°, and 45°. For 90° angles, our rotation routine produces an image in which all combinations of three input pixels produce the same included angles as those produced by the unrotated image. Thus, the three angles used represent a much wider variety of distor-

† For values falling outside [0, 255], the modified gray scale value was rounded to the nearest in-range value.

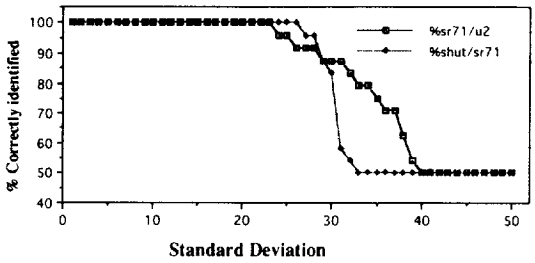

Fig. 13. Tolerance of HONNs to white Gaussian noise introduced as in Fig. 12. Each instantiation of a third-order network designed to be invariant to distortions in scale, translation, and in-plane rotation (one for the SR-71/U-2 problem and one for the Shuttle/SR-71 problem) was tested on 1200 images generated by modifying the 8-bit gray level values of the original images using a Gaussian distribution of random numbers with a mean of 0 and a standard deviation between 1 and 50. The test set consisted of 50 images (with increasing standard deviation from 1 to 50) for the four scales ranging from 70 to 100% in 10% increments, and each rotation of 0°, 30°, and 45°. For 90° angles, our rotation routine produces an image in which all combinations of three input pixels produce the same included angles as those produced by the unrotated image. Thus, the three angles used represent a much wider variety of distorted images than is initially apparent. The noisy images were then binarized and edge detected.

ted images than is initially apparent. The noisy images were then binarized and edge detected. Typical test images, along with their values for standard deviation (σ), scale, and rotation, are shown in Fig. 12.

To have an affect on the processed binary image, the modified gray level value must cross the threshold value. Hence, the amount of noise tolerated will depend on the distribution of gray level values in the original images such that for a given σ, images with low contrast foreground/background values appear noisier than images with high contrast foreground/background values. In our case, the lighting conditions present when the images in Fig. 10 were taken produced dark areas with gray level values of approximately 20 and light areas with gray level values of approximately 220, with decreasing light area values near the edges and where the original model had decals or other anomalies, such as windows. Our threshold value was 128. Thus, our results show the performance of HONNs with white noise in almost ideal conditions.

The results are summarized in Fig. 13. The network performed with 100% accuracy for our test set for a standard deviation of up to 23 on the SR-71/U-2 problem and 26 on the Shuttle/SR-71 problem. For the similar images of the Shuttle and SR-71, the recognition accuracy quickly decreased to 75% at a σ of 30 and to 50% (which corresponds to no better than random guessing) for σ greater than 33. The SR-71/U-2 remained above 75% accuracy up to a σ of 35 (or $\sim 14\%$ of the gray level range) and gradually decreased to 50% at a σ of 40 (or $\sim 16\%$ of the gray level range). If we define "good performance" as greater than 75% accuracy, HONNs have good performance for σ up to 35 (or $\sim 14\%$ of the gray level range) for images with very distinct profiles and σ up to 30 (or $\sim 12\%$ of the gray level range) for images with similar profiles. It should be noted that these results apply only to images with an ideal separation of background/foreground gray levels. For images with a lower contrast, the performance may be quite different.

OCCLUSION

To test the tolerance of HONNs to partial occlusion, the two instantiations (one for the Shuttle/SR-71 problem and one for the SR-71/U-2 problem) of the third-order network built to be invariant to scale, in-plane rotation, and translation as described above were tested on occluded versions of the image pairs for each of four scales ranging from 70 to 100% in 10% increments, and each rotation of 0°, 30°, and 45°. Again, as for white noise, these three angles represent a much wider variety of distorted images than is initially apparent. We started with binary, edge-only images and added automatically-generated occlusions based on four variable parameters: the size of the occlusion, the number of occlusions, the type of occlusion, and the position of the occlusion. Objects used for occlusion were squares with a linear dimension between 1 and 29 pixels. The number of occlusion objects per image varied from one to four, and the randomly chosen type of occlusion determined whether the occlusion objects were added to or subtracted from the original image. Finally, the occlusions were randomly (uniform distribution) placed on the profile of the training images. The test set consisted of 10 samples for each combination of scale, rotation angle, occlusion size, and number of occlusions for a total of 13,920 test images per training image or 27,840 test images per recognition problem. Typical test images are shown in Fig. 14.

As shown in Fig. 15, the performance of HONNs with occluded test images depends mostly on the number and size of occluding objects and to a lesser degree on the similarity of the training images. In the case of the Shuttle/SR-71 recognition problem, the network performed with 100% accuracy for our test set for one 16 pixel occlusion and up to four 10 pixel occlusions. It performed with better than 75% accuracy ("good performance") for up to four 19 pixel occlusions, three 21 pixel occlusions, two 24 pixel occlusions, and one 29 pixel occlusion.

For the SR-71/U-2 problem, the network exhibited good performance for the entire test set but achieved 100% accuracy only for one 4 pixel occlusion and up to four 3 pixel occlusions.

CONCLUSIONS

Higher-order neural networks (HONNs) have been shown to be effective for distinguishing between a set of 2D objects regardless of their position, in-plane rotation, or scale with 100% accuracy on noise-free test images characterized by the built-in distortions. Only

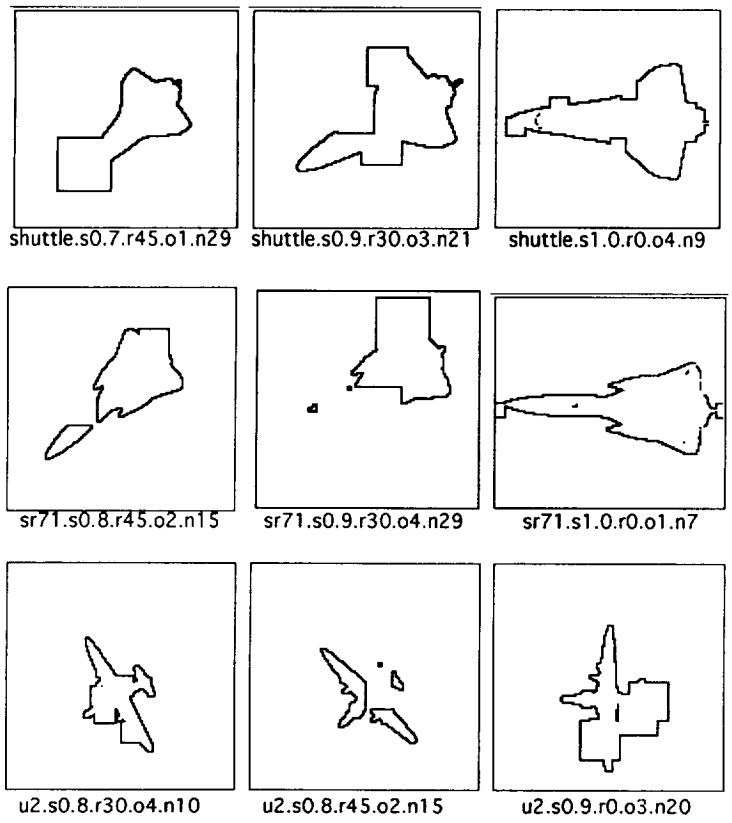

Fig. 14. Typical images used to test the tolerance of HONNs with respect to occlusion. We started with binary, edge-only images and added automatically-generated occlusions based on four variable parameters: the size of the occlusion, the number of occlusions, the type of occlusion, and the position of the occlusion. Objects used for occlusion were squares with a linear dimension between 1 and 29 pixels. The number of occlusion objects per image varied from one to four, and the randomly chosen type of occlusion determined whether the occlusion objects were added to or subtracted from the original image. Finally, the occlusions were randomly (uniform distribution) placed on the profile of the training images. The training set consisted of 10 samples for each combination of scale, rotation angle, occlusion size, and number of occlusions for a total of 13,920 test images per training image or 27,840 test images per recognition problem. Values for number of occlusions and occlusion size range are shown under each typical test image. For example, "s1.0.r30.o3.n20" is an image of scale 100%, rotation 30°, and three 20 pixel occlusions.

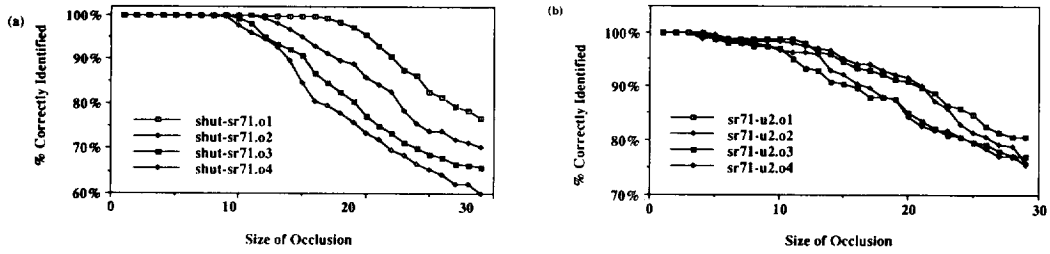

Fig. 15. Tolerance of HONNs with respect to occlusion: (a) Shuttle/SR-71 discrimination problem; (b) SR-71/U-2 discrimination problem. Each graph shows the recognition accuracy for images generated as in the caption of Fig. 14.

one view of each object was required for learning and the network successfully learned to distinguish between all distorted views of the two objects in tens of passes, requiring only minutes on a Sun 3/60. In contrast, other neural network approaches require thousands of passes through a training set consisting of a much larger number of training images yet still achieve only 80–90% accuracy on novel examples.

The major limitation of HONNs is that the size of the input field is limited because of the memory required for the large number of interconnections in a fully connected network. In an $N \times N$ pixel input field, combi-

nations of three pixels can be chosen in N^2-choose-3 ways. Thus, for a 128×128 pixel input field, the number of possible triplet combinations is $\sim 7.3 \times 10^{11}$, a number too great to store on most machines. To circumvent this limitation, we use a coarse coding algorithm which allows a third-order network to be used with a practical input field size while retaining its ability to recognize images which have been scaled, translated, or rotated in-plane.

Further, we explore the tolerance of coarse-coded HONNs to white Gaussian noise and to partial occlusion. We demonstrate that for images with an almost ideal separation of background/foreground gray levels, it takes a great amount of white noise in the gray level images to affect the binary, edge-only images used for training and testing the system to a sufficient degree that the performance of HONNs was seriously degraded. Specifically, we demonstrate that the gray level values must be changed by a standard deviation of at least 10% before the recognition accuracy of HONNs drops below 100%. HONNs continue to show good performance (greater than 75% recognition accuracy) on white noise for a standard deviation up to $\sim 14\%$.

HONNs are also robust with respect to partial occlusion. We trained a third-order network on two sets of images (Shuttle vs. SR-71 and SR-71 vs. U-2) and tested it using numerous occluded images generated automatically by varying the size of the occlusions, the number of occlusions, the type of occlusions, and the location of the occlusions. The amount of occlusion tolerated depends mostly on the size of the occlusion and the number of occlusions, and slightly on the similarity of the training images. On a test set for training images with very similar profiles, HONNs achieved 100% recognition accuracy for one occlusion of $\sim 13\%$ of the input field size and four occlusions of $\sim 7\%$ of the input field size. They showed good performance for one occlusion of $\sim 23\%$ of the input field size or four occlusions of $\sim 15\%$ of the input field size each. On the test set for training images with very different profiles, HONNs achieved 100% recognition accuracy for up to four occlusions of $\sim 2\%$ of the input field size and continued to show good performance for up to four occlusions of $\sim 23\%$ of the input field size each.

The results for white Gaussian noise reflect the performance of coarse-coded third-order neural networks on training images with high contrast gray level. For test images with lower contrast gray levels, we expect the tolerance of the network to white noise to decrease while for test images with even higher contrast gray levels, we expect the tolerance to increase. For partial occlusions, unlike for white noise, the initial gray levels do not have as great an effect on the performance and thus can be generalized more readily.

All of the above results focus on the capabilities of HONNs for position, scale, and rotation-invariant object recognition for 2D shapes or 3D objects constrained to rotate within a plane. In order to robustly recognize an object which has been rotated out of plane, more training images are required. However, since the 2D recognition task is a component of the 3D recognition task, we believe HONNs will outperform other methods even in the broader domain.

REFERENCES

1. J. Mantas, Methodologies in pattern recognition and image analysis—a brief survey, *Pattern Recognition* **20**, 1–6 (1987).
2. C. G. Perrott and G. C. Hamey, Object recognition, a survey of the literature, Macquarie Computing Reports, Report No. 91-0065C, School of MPCE, Macquarie University, Australia (1991).
3. D. E. Rumelhart, G. E. Hinton and R. J. Williams, Learning internal representations by error propagation, *Parallel Distributed Processing*, Vol. 1, Chap. 8. MIT Press, Cambridge, Massachusetts (1986).
4. S. E. Troxel, S. K. Rogers and M. Kabrisky, The use of neural networks in PSRI recognition, *Proc. Joint Int. Conf. on Neural Networks*, San Diego, California, pp. 593–600 (1988).
5. G. L. Giles and T. Maxwell, Learning, invariances, and generalization in high-order neural networks, *Appl. Optics* **26**, 4972–4978 (1987).
6. G. L. Giles, R. D. Griffin and T. Maxwell, Encoding geometric invariances in higher-order neural networks, *Neural Information Processing Systems, American Institute of Physics Conf. Proc.*, pp. 301–309 (1988).
7. M. B. Reid, L. Spirkovska and E. Ochoa, Simultaneous position, scale, and rotation invariant pattern classification using third-order neural networks, *Int. J. Neural Networks* **1**, 154–159 (1989).
8. R. Rosenfeld and D. S. Touretzky, A survey of coarse-coded symbol memories, *Proc. 1988 Connectionist Models Summer School*, Carnegie Mellon University, pp. 256–264 (1988).
9. J. Sullins, Value cell encoding strategies, Technical Report TR-165, Computer Science Department, University of Rochester, Rochester, New York (1985).
10. L. Spirkovska and M. B. Reid, Applications of higher-order neural networks in the PSRI object recognition domain, *Fuzzy, Holographic, Invariant and Parallel Intelligence: The Sixth Generation Breakthrough*, B. Soucek and the IRIS Group, eds. Wiley, New York (1992).

About the Author—LILLY SPIRKOVSKA received the B.S. degree in mathematics and computer science from the University of Denver in November 1984, and the M.S. degree in computer science from the University of California at Berkeley in May 1986. Ms Spirkovska's research in the Computational Systems Research Branch at NASA Ames Research Center emphasizes machine vision techniques. She is a member of the Association for Computing Machinery (ACM) and the International Neural Network Society (INNS) as well as the honor societies Phi Beta Kappa and Pi Mu Epsilon. She has published numerous articles in the area of neural pattern recognition, including two book chapters.

About the Author—MAX REID received the B.S. degree in electrical engineering and computer science with Special Honors from the University of Colorado, Boulder, in 1983, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, California, in 1986 and 1988, respectively. Dr Reid is currently the Photonics Group Leader in the Computational Systems Research Branch at NASA Ames Research Center, Mountain View, California. His research includes optical and neural pattern recognition techniques, optical matrix processors, and optical implementations of neural networks. He is a member of the Institute of Electronic Electrical Engineers, the Optical Society of America, the American Institute of Physics, and the International Neural Network Society. He has published over 30 journal and conference articles in the areas of optical processors, lasers, and neural networks.